

An Inquiry Into PBNM System Performance Required For Massive Scale Telecommunication Applications

Shane Magrath
Doctorate of Philosophy (Engineering)
Year of Submission: 2006

Institute of Information and Communications Technology
School of Engineering
University of Technology, Sydney
Australia



Contents

1	Introduction	14
1.1	Problem Statement	14
1.2	Aim	14
1.3	Overview of the Study	15
2	Background	17
2.1	Introduction	17
2.2	Policy-Based Network Management	17
2.2.1	Ontology	17
2.2.2	Policy Specification and Languages	21
2.2.2.1	Ponder	22
2.2.2.2	<i>PDL</i>	23
2.2.2.3	P-CIM	24
2.2.3	Policy Conflict	26
2.2.4	Policy Refinement	27
2.2.5	Policy Systems	27
2.2.5.1	IETF, COPS and P-CIM	27
2.2.5.2	Ponder	29
2.2.5.3	APMS	32
2.2.6	PBNM Applications	32
2.2.6.1	QoS	32
2.2.6.2	VPN	32
2.2.6.3	Access Control	33
2.2.6.4	Service Level Management	33
2.2.7	Operational Studies	33
2.3	Expert Systems	34
2.3.1	Single Agent Systems	35
2.3.2	Pattern Matching Algorithms	35
2.3.3	Performance and Benchmarking	38
2.4	Conclusion	38
2.4.1	PBNM Status	38
2.4.2	IETF	39
2.4.3	PBNM System Performance	40

3	Theory	42
3.1	Policy System Architectures	42
3.1.1	Important Differences	43
3.2	Performance	44
3.2.1	Concepts	44
3.2.2	Analytical Model	45
3.2.3	Performance Requirements	47
3.3	The Rete Algorithm	49
3.3.1	Description	50
3.3.2	Atomic Operators	51
3.3.3	Expression Parsing	57
3.4	The MatchBox Algorithm	58
3.4.1	Description	58
3.4.2	MatchBox Operations	61
3.4.3	PDN	62
3.4.4	Negation	64
3.4.5	Problems with MatchBox	64
3.5	The JukeBox Algorithm	65
3.5.1	Description	65
3.5.2	Bindspace	67
3.5.3	Relational Subspace Intersections	70
3.5.4	Correct MatchBox and BindSpace Operations	72
3.6	Conclusion	77
4	Experimental Design	78
4.1	Engineering Epistemology	78
4.2	Research Methods	79
4.3	Aim	81
4.4	Method	81
4.4.1	Research Hypotheses	81
4.4.2	Experimental Apparatus	82
4.4.3	Procedure	88
4.4.4	Sources and Control of Experimental Errors	88
4.5	H1: Dominance Hypothesis	91
4.5.1	Refinement	91
4.5.2	Operational Hypothesis ($H_{1,1}$) - <u>Low Level</u> Policy Test	91
4.5.3	Operational Hypothesis ($H_{1,2}$) - <u>High Level</u> Policy Test	91
4.5.4	Justification	91
4.5.5	W&M Testing Procedure	92
4.5.6	B1 Test Specification	92
4.5.7	B2 Test Specification	94
4.6	H2: Complexity Hypothesis	96
4.6.1	Refinement	96

4.6.2	Operational Hypothesis ($H_{2.1}$)	96
4.6.3	Justification	96
4.6.4	W&M Testing Procedure	96
4.6.5	B3 Test Specification	97
4.6.6	B4 Test Specification	99
4.7	H3: Bindspace Hypothesis	100
4.7.1	Refinement	100
4.7.2	Operational Hypothesis ($H_{3.1}$)	100
4.7.3	Justification	100
4.7.4	W&M Testing Procedure	101
4.8	Conclusion	104
5	Results	106
5.1	Dominance Hypothesis Results	106
5.1.1	Description	106
5.1.2	Presentation	107
5.1.3	Analysis	111
5.1.3.1	H1.1 Results	111
5.1.3.2	H1.2 Results	112
5.1.4	Interpretation	113
5.2	Complexity Hypothesis Results	115
5.2.1	Description	115
5.2.2	Presentation	116
5.2.3	Analysis	120
5.2.3.1	H2a Results	120
5.2.3.2	H2b Results	120
5.2.4	Interpretation	121
5.3	Bindspace Hypothesis Results	123
5.3.1	Description	123
5.3.2	Presentation	124
5.3.3	Analysis	127
5.3.3.1	Low Dimensional Bindspace (B5)	127
5.3.3.2	High Dimensional Bindspace (B6)	127
5.3.4	Interpretation	128
5.4	Conclusion	128
6	Discussion and Conclusions	130
6.1	Discussion of Results	130
6.1.1	Findings on Inferencing Performance	130
6.1.2	Findings on Conjunctive Matching	131
6.1.3	Findings on Overall PBNM System Performance	132
6.1.4	Findings on PBNM System Design	133
6.2	Conclusions	133
6.2.1	Aim One	134

6.2.2	Aim Two	134
6.2.3	Aim Three	134
6.2.4	Aim Four	135
6.2.5	Limitations to the Research and its Findings	135
6.3	Directions for Future Work	136
Bibliography		138
A The Rete Algorithm		146
A.1	Introduction	146
A.2	Discrimination Networks	146
A.2.1	Informal Operational Description	146
A.2.2	PDN Generation	148
A.3	Data and Tokens	148
A.3.1	Data	148
A.3.2	Tokens	148
A.4	UML Class Diagram	149
A.4.1	PDN Nodes	149
A.5	Root Node	149
A.6	Alpha Nodes	150
A.7	Beta Nodes	150
A.7.1	Beta Memory Operation	151
A.8	Conjunct Nodes	151
A.9	Negation Nodes	155
A.9.1	Negation Node Operation	155
A.9.2	Negation Involving Join Variables	159
A.10	Variable Scope	159
A.11	Terminal Nodes	161
A.12	Miscellaneous Nodes	162
A.13	ActivationSet	162
A.14	Implementation Differences from Classical RETE	162
A.14.1	ActivationSet	162
A.14.2	Beta Nodes and Memories	163
B The JukeBox Algorithm		164
B.1	Introduction	164
B.2	MatchBox Concepts	164
B.3	Problems with MatchBox	166
B.4	JukeBox Concepts	166
B.5	JukeBox Implementation	167
B.5.1	Compilation Support and the JoinMap Structure	168
B.5.2	Sparse Storage of Matchboxes	169
B.5.3	Relational Subspace Data	170
B.6	MatchBox Operations	170

B.7	JukeBox and Negation	170
B.8	Parenthetical Expressions	172
C	The JitterBug System	174
C.1	JitterBug Measures	174
C.2	Event Listeners	176
C.2.1	Function	176
C.2.2	Patterns	177
C.2.3	Threading Model	177
C.2.4	Management	178
C.3	Event Management	179
C.3.1	Function	179
C.3.2	Patterns	179
C.3.3	Threading Model	179
C.3.4	Management	179
C.3.5	Implementation	179
C.4	Policy Inferencing Management	182
C.4.1	Function	182
C.4.2	Patterns	182
C.4.3	Threading Model	183
C.4.4	Management	183
C.4.5	Implementation	183
C.5	Executor Management	186
C.5.1	Function	186
C.5.2	UML	187
C.5.3	Threading Model	187
C.5.4	Management	187
C.5.5	Implementation	187
C.5.5.1	Executor	187
C.5.5.2	Library	188
C.5.5.3	Functions	189
C.6	OAM	191
C.6.1	Base TepMBeans Interface	191
C.6.1.1	Attributes	191
C.6.1.2	Operations	192
C.6.1.3	JMX Naming	192
D	The Jive! Language and Compiler	193
D.1	Example	193
D.2	Compiler Design	194
D.2.1	SableCC	194
D.3	Jive! Grammar	195
D.4	Unit Tests	198
D.4.1	Introduction	198

D.4.2	Unit Tests	198
E	Geneva: The Elvin Event Generator	214
E.1	Purpose	214
E.2	Operation	214
E.3	Design	215
F	Session Traffic Generation Model	217
G	Experimental Output Files	221
G.1	TepData.dat	221
G.2	TepOutput.dat	221
G.3	TepLatency.dat	222
G.4	Matlab .M files	222
G.4.1	TepJitterBug.m	222
G.4.2	TepH1.m	223
G.5	Measurement Procedure	225

List of Figures

1.1	Suggested Reading Path	16
2.1	PBNM Research Program	18
2.2	Simplified UML of P-CIM	25
2.3	COPS Architecture	28
2.4	Ponder Structural Architecture	30
2.5	Rule Discrimination Structure	36
3.1	Policy System Architecture	43
3.2	Simple Queue - One Server	45
3.3	Response Time of a Single Server for Various Loads	49
3.4	Example PDN and Policy Condition	50
3.5	Conjunctive Matching Example (AND node)	54
3.6	MatchBox vs Rete PDNs	63
3.7	MatchBox vs Rete from Lee & Cheng	65
3.8	Example: Forming a 2-Dimensional Bindspace	66
3.9	Example 2-Dimensional Sparse Store	71
4.1	Experimental Configuration	82
4.2	Geneva	84
4.3	JitterBug GUI	87
4.4	PDNs for B1 and B2 Tests	93
4.5	PDNs for B3 and B4 Tests	98
4.6	PDNs for H3 Tests	102
5.1	PDNs for B1 and B2 Tests	107
5.2	H1 PDN Latency Results	108
5.3	H1 Total System Latency Results	109
5.4	H1 PDN Occupancy Results	110
5.5	PDNs for B3 and B4 Tests	116
5.6	H2 PDN Latency Results	117
5.7	H2 Total System Latency Results	118
5.8	H2 PDN Percentage Occupancy	119
5.9	PDN for B6 Tests	124

5.10	H3 Low-Level (B5) PDN Latency	125
5.11	H3 High-Level (B6) PDN Latency	126
A.1	Example PDN	147
A.2	UML Class Diagrams of PDN Nodes	149
A.3	Conjunct (And) Node PDN	152
A.4	Conjunction with Variable Number of Join Tests	153
A.5	PDN For Simple Negation	156
A.6	Complex Negation	157
A.7	Variable Scope and Sub-Expressions	160
A.8	Complex Negation Example	161
B.1	Two dimensional bindspace example	165
B.2	Example JukeBox PDN Call Graph	168
B.3	Example 2-Dimensional Sparse Store	170
B.4	Simple Negation PDN	172
C.1	JitterBug System	175
C.2	Listener Relay Pattern	178
C.3	Listener JMX Management Pattern	178
C.4	Monitor Pattern	180
C.5	Event Sub-System Design	181
C.6	Policy Sub-System	182
C.7	JMX OAM Design for the Policy Sub-System	184
C.8	Class Diagram for the Executor	187
C.9	JMX Architecture	191
E.1	Geneva Configuration Console	215
E.2	Geneva UML Class Diagram	216

List of Tables

2.1	Benchmark Applications Maintained by Miranker ([84])	38
3.1	Offered Load of DIFFSERV Traffic for Variously Sized Networks	48
5.1	H1 Statistics 111	
5.2	H2 Statistics 120	

Certificate of Authorship/Originality

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate

Statement of Contribution

Contributions to Knowledge

We have proposed and tested three hypotheses about policy system performance in massive-scale applications:

- The *dominance* hypothesis which claims “PBNM system performance in large domains is dominated by the latency contribution of the inferencing sub-system”.
- The *complexity* hypothesis which claims “Policy *complexity* as measured by PDN depth is a more significant factor in degrading performance than the *number* of policies in the system, in large domains.”
- The *bindspace* hypothesis which claims “A novel *bind-space* conjunctive matching algorithm known as JukeBox will produce superior performance when compared to the current state of the art *assertion-space* algorithm known as Rete, for large domains.”

All three hypotheses were *refuted* and the implications of these findings worked out as new knowledge about policy system performance.

In the course of examining these hypotheses, the following contributions were also made:

- Policy server *IO*, both network and file, is a significant performance issue and serves to limit the possible achievable performance of the system. This finding is at variance with the literature which assumes the inferencing algorithm is the cause of poor system performance.
- The *fanout* (outdegree) of a node within the policy discrimination network is a significant performance issue to policy server performance and is more important than the total number of policies installed in the system, their apparent complexity and whether conjunctive matching is employed or not.
- In general, it does not appear possible that a pure bindspace conjunctive matching algorithm can outperform an assertion-space algorithm such as hashed-equality Rete. This finding is at variance with the literature.
- We have discovered, documented and implemented a pure bindspace, unconstrained inferencing algorithm for conjunctive match which we called JukeBox. This algorithm is based on Perlin’s MatchBox algorithm but does not have the significant limitations of his algorithm.

- We have identified and corrected an important error in Perlin's original and published Match-Box algorithm. This error prevented the matchbox operations from being generally correct.
- We have discovered, documented and implemented a method for using a binary-tree abstract data type as an efficient sparse store of N-dimensional data.
- We have provided a benchmarking procedure for measuring the performance of policy systems in a visible and repeatable manner. This process allows for the characterisation of the policy server's performance in terms of average latency and its variance.
- We have proposed the simple heuristic of $\lambda = 0.6\mu$ as a useful dimensioning tool for policy system performance planning. This relation is based on a simple consideration of the Pollaczek-Khinchine result from Queueing theory.
- We have provided an open experimental policy system and tool-set that facilitates the research into policy system performance. This tool-set consists of
 - (a) the *Jive!* language for policy specification,
 - (b) an optimising *compiler* for Jive! policies,
 - (c) the *JitterBug* policy enforcement system and
 - (d) *Geneva* - a configurable policy event generator.

Related Publications

- *Policy Interoperability and Network Autonomics*, Workshop on Autonomic Communications 2004 (WAC2004), Berlin, 2004, Springer-Verlag LNCS3457
- *Autonomics in Telecommunications Service Activation*, IEEE Workshop on Autonomic Communication for Evolvable Next Generation Networks, 7th International Symposium on Autonomous Decentralized Systems, 2005
- *Scalable Policy Enforcement and PBNM Benchmarking*, Ninth IFIP/IEEE International Symposium on Integrated Network Management (IM 2005), 2005 - poster session
- *Using Policy Mechanisms to Implement Autonomic Behaviour in Telecommunication Service Activation Processes*, 2006 IEEE/IFIP Network Operations & Management Symposium - accepted as a poster.

Abstract

PBNM systems have been proposed as a feasible technology for managing massive scale applications including telecommunication service management. What is not known is how this class of system performs under carrier-scale traffic loads. This research investigates this open question and concludes, subject to the considerations herein, this technology can provide services to large scale applications.

An in depth examination of several inferencing algorithms is made using experimental methods. The inferencing operation has been implicated as the major source of performance problems in rule based systems and we examine this. Moreover, these algorithms are of central importance to current and future context-aware, pervasive, mobile services. A novel algorithm, JukeBox, is proposed that is a correct, general and pure bindspace conjunctive match algorithm. It is compared to the current state of the art algorithm - Rete. We find that Rete is the superior algorithm when implemented using the hashed-equality variant.

We also conclude that IO is an important cause of PBNM system performance limitations and is perhaps of more significance than the implicated inferencing operations. However, inferencing can be a bottleneck to performance and we document the factors associated with this.

We describe a generally useful policy system benchmarking procedure that provides a visible, repeatable and measurable process for establishing a policy server's service rate characteristics. The service rate statistics, namely μ and σ , establish the limitations to policy system throughput. Combined with the offered traffic load to the server, using the statistic λ , we can provide a complete characterisation of system performance using the Pollaczek-Khinchine function. This characterisation allows us to make simple design and dimensioning heuristics that can be used to rate the policy system as a whole.